Week 2 - Lab

Clarissa Boyajian

2022-04-12

Contents

Query NY Times

The code below creates my query to the New York Times for the terms "urban forest":

```
# create query parts
api_key <- "qTBZr0Mz2Ak8icY1OZTCm0kfxdze9Gnn"
term <- "urban+forest" # use + to string together separate words
begin_date <- "20210401" # YYYYMMDD
end_date <- "20220401" # YYYYMMDD

# construct the query url using API operators
baseurl <- paste0("http://api.nytimes.com/svc/search/v2/articlesearch.json?q=", term,
                  "&begin_date=", begin_date,
                  "&end_date=", end_date,
                  "&facet_filter=true&api-key=", api_key,
                  sep = "")
```

The code below pulls 15 pages of query results and combines into one dataframe:

```
# this code allows for obtaining multiple pages of query results
# (each search maxes out at 10 pages for NYTimes)
initialQuery <- fromJSON(baseurl)

maxPages <- round((initialQuery$response$meta$hits[1] / 10) - 1)

pages <- list()
for(i in 0:maxPages){
  nytSearch <- fromJSON(paste0(baseurl, "&page=", i), flatten = TRUE) %>%
```

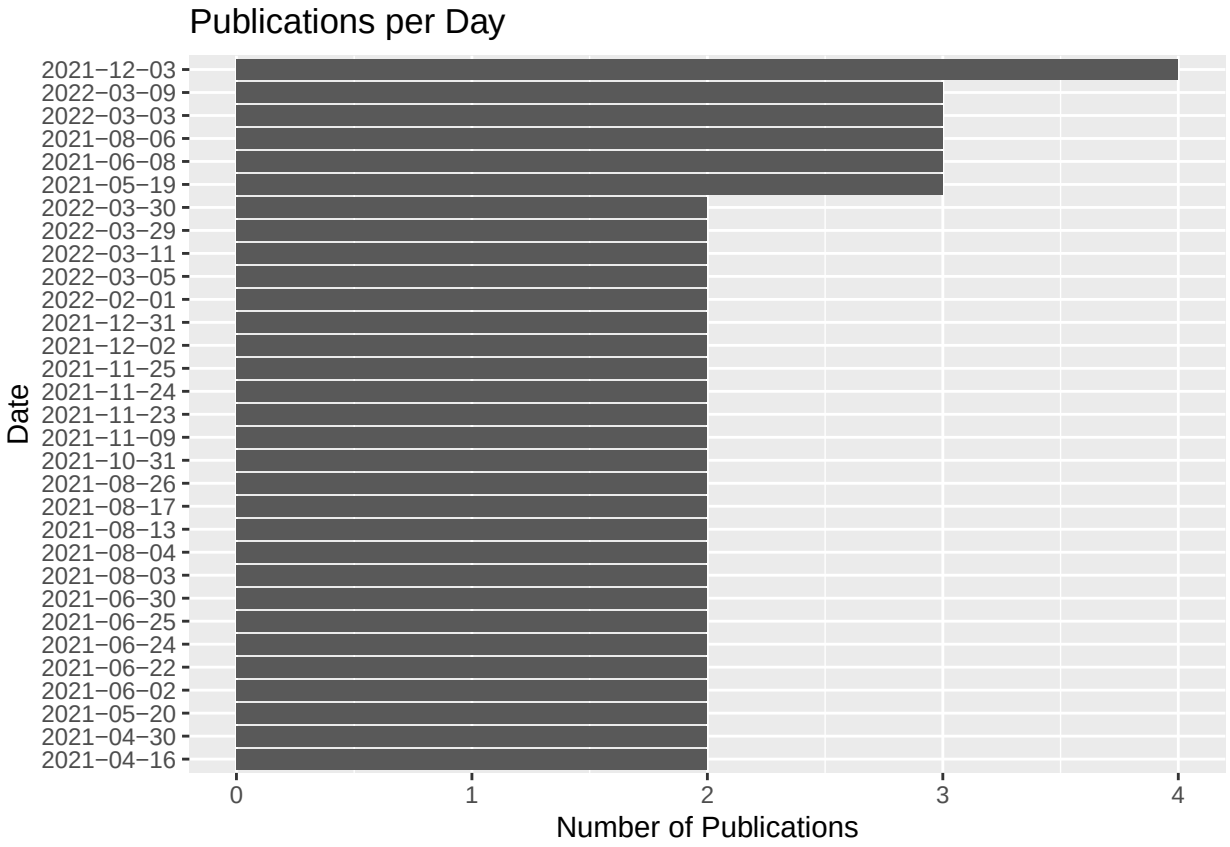
```
    data.frame()
  message("Retrieving page ", i)
  pages[[i + 1]] <- nytSearch
  Sys.sleep(6) # keeps you from hitting limit for API
}
class(nytSearch)

# need to bind the pages and create a tibble from nytDa
nytDat <- rbind_pages(pages)

saveRDS(object = nytDat,
        file = here::here("labs/nytDat.rds"))
```

```
# read query back in (for speed when knitting)
nytDat <- readRDS(file = here::here("labs/nytDat.rds"))
```

Publications per day plot - Lead Paragraphs

The code below creates a plot of publications per day:

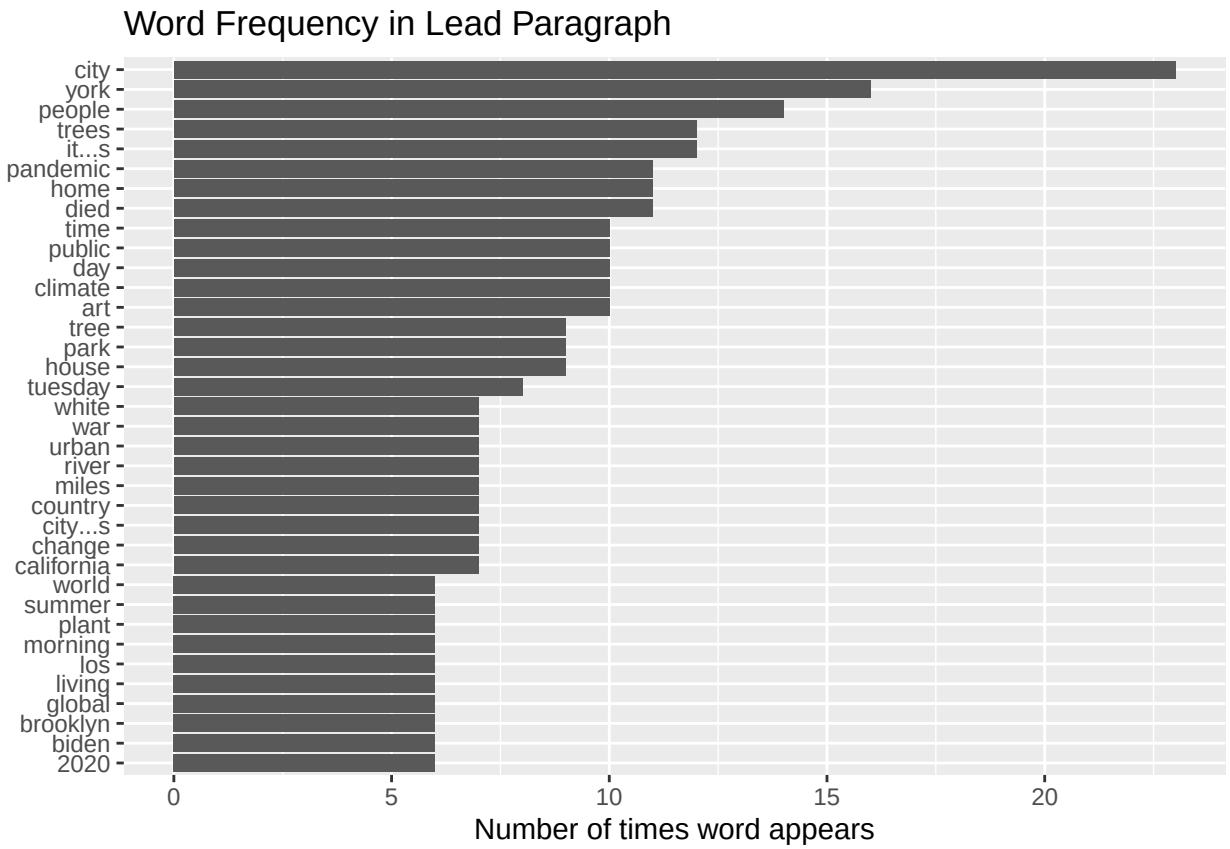
```
nytDat %>%
  # replace "T." with "" - remove time but leave dates
  mutate(pubDay = gsub("T.*", "", response.docs.pub_date)) %>%
  group_by(pubDay) %>%
  summarise(count = n()) %>%
  filter(count >= 2) %>%
  ggplot() +
  geom_bar(aes(x = reorder(pubDay, count),
               y = count),
           stat = "identity") +
  coord_flip() +
  labs(title = "Publications per Day",
       y = "Number of Publications",
       x = "Date")
```



Word frequency plot - Lead Paragraphs

The code below creates a word frequency plot for lead paragraphs with only stopwords removed:

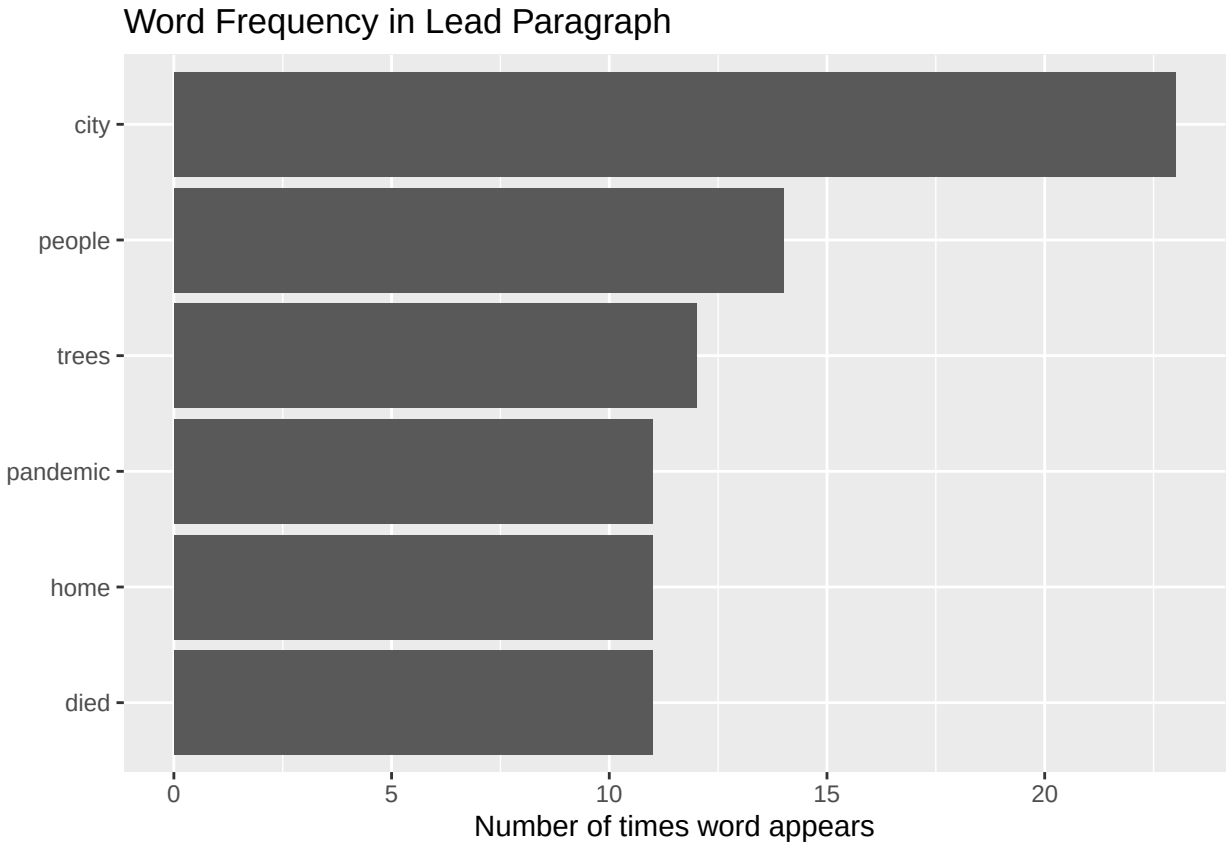
```
paragraph <- names(nytDat)[6] # call the 6th column ("response.doc.lead_paragraph")

data(stop_words) # pull data object from `tidytext` package

# convert from text to `tidytext` format
tokenized <- nytDat %>%
  # take paragraphs in and un-nest to word level (1 row for each word in paragraph)
  unnest_tokens(word, paragraph) %>%
  # remove all rows that match a stopword
  anti_join(y = stop_words, by = "word")

# plot of words and # of times appear (after removing stopwords)
tokenized %>%
  count(word, sort = TRUE) %>%
  filter(n > 5) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word)) +
  geom_col() +
  labs(y = NULL,
       x = "Number of times word appears",
       title = "Word Frequency in Lead Paragraph")
```



The code below creates a word frequency plot for lead paragraphs with additional words remove or stemmed as needed:

```
# remove and stem words
clean_tokens <- str_remove_all(string = tokenized$word,
                               pattern = "[:digit:]")
clean_tokens <- str_remove_all(string = clean_tokens,
                               pattern = ",")
clean_tokens <- str_remove_all(string = clean_tokens,
                               pattern = "'s")
clean_tokens <- str_remove_all(string = clean_tokens,
                               pattern = "^s$")
clean_tokens <- str_remove_all(string = clean_tokens,
                               pattern = "it's")
clean_tokens <- str_remove_all(string = clean_tokens,
                               pattern = "york")
# put the cleaned tokens into the `tokenized` df `clean` column
tokenized$clean <- clean_tokens

# remove the empty strings
tib <-subset(tokenized, clean != "")

# reassign
tokenized <- tib

# new plot with more words removed
```

```
tokenized %>%
  count(clean, sort = TRUE) %>%
  filter(n > 10) %>% # illegible with all the words displayed
  mutate(clean = reorder(clean, n)) %>%
  ggplot(aes(n, clean)) +
  geom_col() +
  labs(y = NULL,
       x = "Number of times word appears",
       title = "Word Frequency in Lead Paragraph")
```

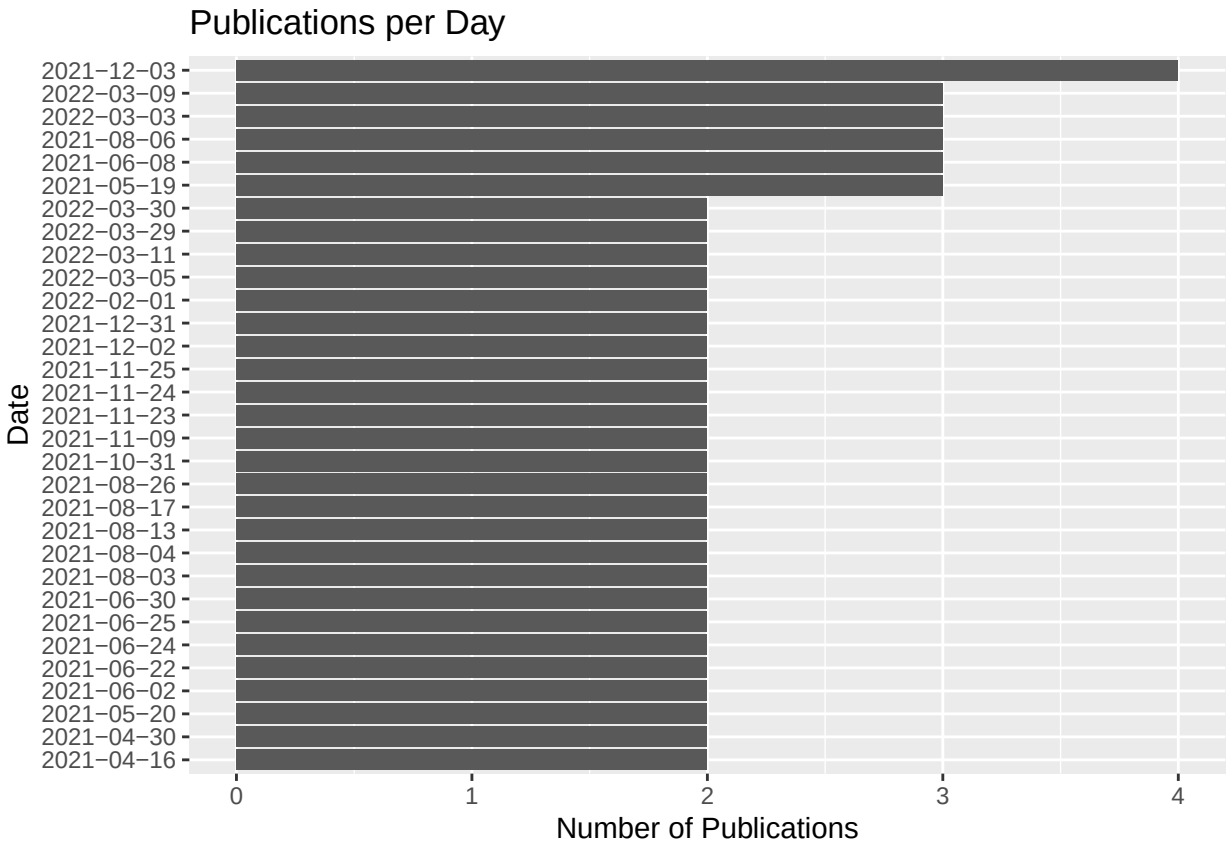


Publications per day plot - Headlines

The code below creates a plot of publications per day:

```
nytDat %>%
  # replace "T." with "" - remove time but leave dates
  mutate(pubDay = gsub("T.*", "", response.docs.pub_date)) %>%
  group_by(pubDay) %>%
  summarise(count = n()) %>%
  filter(count >= 2) %>%
  ggplot() +
  geom_bar(aes(x = reorder(pubDay, count),
               y = count),
           stat = "identity") +
```

```
  coord_flip() +
  labs(title = "Publications per Day",
       y = "Number of Publications",
       x = "Date")
```

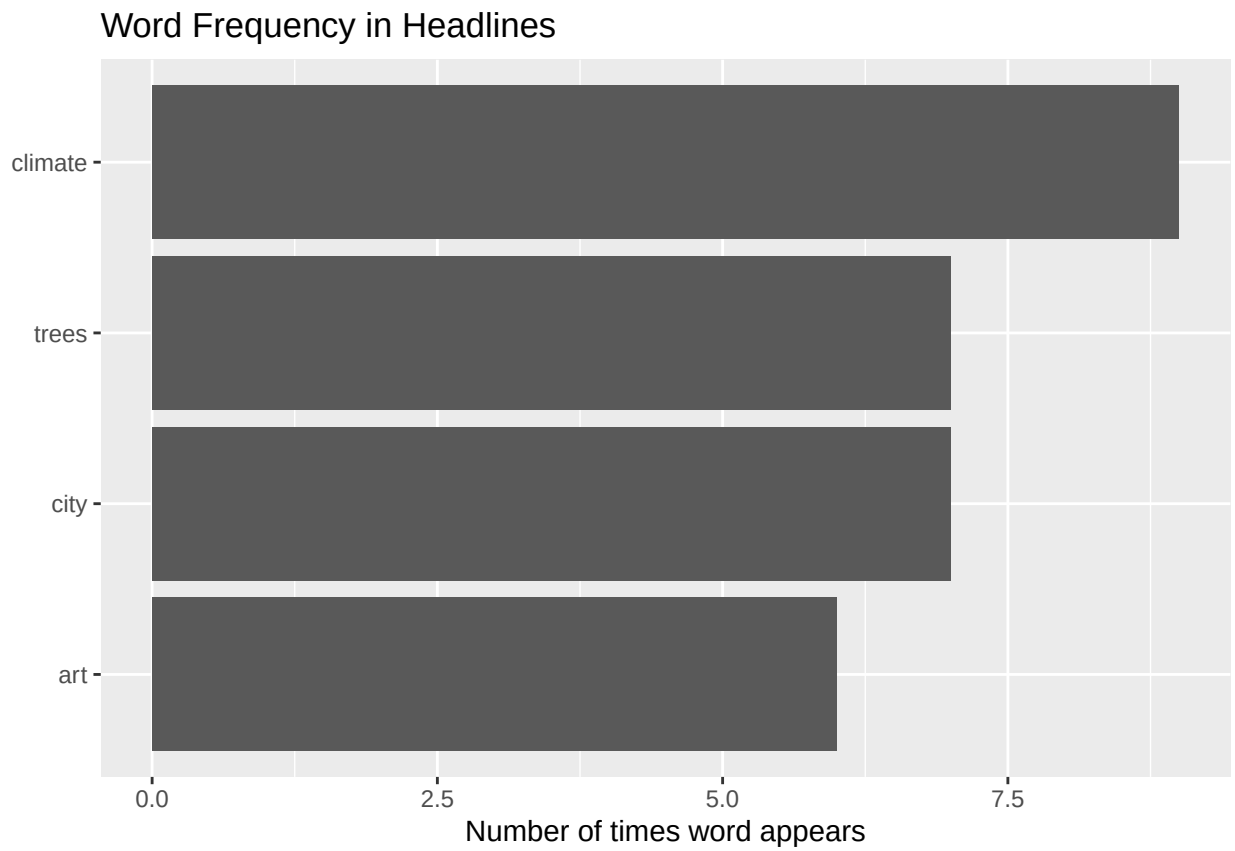


Word frequency plot - Headlines

The code below creates a word frequency plot for headlines with additional words remove or stemmed as needed:

```
headline <- names(nytDat)[21] # call the 6th column ("response.doc.lead_paragraph")

# convert from text to `tidytext` format
tokenized_headline <- nytDat %>%
  # take headlines in and un-nest to word level (1 row for each word in headline)
  unnest_tokens(word, headline) %>%
  # remove all rows that match a stopword
  anti_join(y = stop_words, by = "word")

# remove and stem words
clean_tokens <- str_remove_all(string = tokenized_headline$word,
                               pattern = "[:digit:]")
clean_tokens <- str_remove_all(string = clean_tokens,
                               pattern = ",")
```

```r
clean_tokens <- str_remove_all(string = clean_tokens,
                               pattern = "'s")
clean_tokens <- str_remove_all(string = clean_tokens,
                               pattern = "^s$")
clean_tokens <- str_remove_all(string = clean_tokens,
                               pattern = "it's")
clean_tokens <- str_remove_all(string = clean_tokens,
                               pattern = "york")
# put the cleaned tokens into the `tokenized_headline` df `clean` column
tokenized_headline$clean <- clean_tokens

# remove the empty strings
tib_headlines <-subset(tokenized_headline, clean != "")

# reassign
tokenized_headline <- tib_headlines

# create plot
tokenized_headline %>%
  count(clean, sort = TRUE) %>%
  filter(n > 5) %>% # illegible with all the words displayed
  ggplot(aes(x = n,
             y = reorder(clean, n))) +
  geom_col() +
  labs(y = NULL) +
  labs(y = NULL,
       x = "Number of times word appears",
       title = "Word Frequency in Headlines")
```



Plot comparison discussion

The word frequency plots for the lead paragraphs and the headlines overlap in many ways. It is interesting that the more positive words from the lead paragraph plot (such as "tree" and "city") show up in the headlines, whereas the less positive words (such as "pandemic" and "died") do not. It is also interesting that "climate" is the top word in the headlines but does not show up at all in the lead paragraphs. Finally, I wonder why "art" is showing in headlines but not in the lead paragraphs. If I were to dig into this project further, I may want to refine my search terms to exclude queries that are less relevant to urban forests.